UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/807,498	03/24/2004	David John Butcher	550-541	4617

23117 7590 01/08/2008
NIXON & VANDERHYE, PC
901 NORTH GLEBE ROAD, 11TH FLOOR
ARLINGTON, VA 22203

EXAMINER	
LI, AIMEE J	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
01/08/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

mN

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

MAILED

JAN 08 2008

Technology Center 2100

Application Number: 10/807,498
Filing Date: March 24, 2004
Appellant(s): BUTCHER ET AL.

Stanley C. Spooner (Reg. No. 27,393)
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 15 October 2007 appealing from the Office action mailed 13 December 2006.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

No amendment after final has been filed.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

WITHDRAWN REJECTIONS

The following grounds of rejection are not presented for review on appeal because they have been withdrawn by the examiner. The rejections of claims 31-60 under 35 USC §112, first and second paragraphs are withdrawn.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6,363,522	Click, Jr.	3-2002

5,430,862	Smith et al.	7-1995
5,590,294	Mirapuri et al.	12-1996

Tanenbaum, Andrew S. "Structured Computer Organization". Second Edition. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., ©1984. Pages 10-12.

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1, 6-11, 14-15, 16, 21-26, 29-30, 31, 36-41, 44-45, 46, 51-56, and 59-60 are rejected under 35 U.S.C. 103(a) as being unpatentable over Click, Jr. et al., U.S. Patent Number 6,363,522 (herein referred to as Click) in view of Smith et al., U.S. Patent Number 5,430,862 (herein referred to as Smith).

Referring to claims 1, 16, and 31, taking claim 1 as exemplary, Click has taught an apparatus for processing data comprising:

Wherein said instruction decoder, in response to a memory access instruction,

Compares a base register value, stored within a base register specified by a base register field of said memory access instruction, with a predetermined null value (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7); and, If said base register value matches said predetermined null value, then said decoder triggers branching to execution of a null value exception handler (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7).

Click has not explicitly taught

Processing logic operable to perform data processing operations; and

An instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions.

However, Click has taught that the memory access instructions are executed on a processor, as shown in Figure 5, but provides no details about the "N Processor". Smith has taught

Processing logic operable to perform data processing operations (Smith column 2, line 53 to column 4, line 43; Figure 1; and Figure 2); and

An instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions (Smith column 2, line 53 to column 4, line 43; Figure 1; and Figure 2).

A person of ordinary skill in the art at the time the invention was made, and as taught by Smith, would have recognized that the processor of Smith increases performance and compatibility by allowing instructions from multiple instruction sets execute (Smith column 1, lines 36-39) and reducing the need to access off-chip memory (Smith column 2, lines 13-24).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the processor of Smith in the device of Click to increase processor performance and compatibility.

Claims 16 and 31 have similar limitations to claim 1 and are rejected for similar reasons. The only differences are that claim 16 is for a method and claim 31 is for a computer program product.

Referring to claim 46, Click has taught a computer readable medium containing a computer program product comprising a computer program having native program instructions, said native program instructions comprising,

> In response to memory access instruction decodable by said instruction decoder to control said processing logic:
>
> > Comparing a base register value stored within a base register specified by a base register field of said memory access instruction with a predetermined null value (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7); and,
> >
> > If said base register value matches said predetermined null value, then triggering branching to execution of a null value exception handler (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7).

Click has not explicitly taught a computer program product including a computer program operable to translate non-native program instructions to form native program instructions directly decodable by an apparatus for processing data having processing logic operable to perform data processing operations and an instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions. However, Click has taught that the memory access instructions are executed on a processor, as shown in Figure 5, but provides no details about the "N Processor". Smith has taught a computer program product including a computer program operable to translate non-native program instructions to form native program instructions directly decodable by an apparatus for processing data having processing logic operable to perform data processing operations and an instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions. (Smith column 2, line 53 to column 4, line 43; Figure 1; and Figure 2). A person of ordinary skill in the art at the time the invention was made, and as taught by Smith, would have recognized that the processor of Smith increases performance and compatibility by allowing instructions from multiple instruction sets execute (Smith column 1, lines 36-39) and reducing the need to access off-chip memory (Smith column 2, lines 13-24). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the processor of Smith in the device of Click to increase processor performance and compatibility.

Referring to claims 6, 21, 36, and 51, taking claim 6 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 1, wherein said null value exception handler is operable to determine if said memory access instruction attempting to access a location corresponding to a null value corresponds to emulation of a non-native program instruction that is not directly decodable by said instruction decoder attempting to make a memory access using a null value (Click column 1, line 66 to column 2, line 47; column 3, line 48 to column 4, line 7; and column 6, lines 45-67). Claims 21, 36, and 51 have similar limitations to claim 6 and are rejected for similar reasons. The only differences are that claim 21 is for a method and claims 36 and 51 are for a computer program product.

Referring to claims 7, 22, 37, and 52, taking claim 7 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 1, wherein said null value exception handler is operable to determine if said memory access instruction attempting to access a location corresponding to a null value corresponds to an error in operation of a virtual machine computer program operable to translate non-native program instructions that are not directly decodable by said instruction decoder into native program instructions that are directly decodable by said instruction decoder (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 22, 37, and 52 have similar limitations to claim 7 and are rejected for similar reasons. The only differences are that claim 22 is for a method and claims 37 and 52 are for a computer program product.

Referring to claims 8, 23, 38, and 53, taking claim 8 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 6, wherein said non-native program instructions are machine independent program instructions (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 23, 38, and 53 have similar limitations to claim 8 and are rejected for similar reasons. The only differences are that claim 23 is for a method and claims 38 and 53 are for a computer program product.

Referring to claims 9, 24, 39, and 54, taking claim 9 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 8, wherein said machine independent program instructions are one of:

Java bytecodes (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7);

MSIL bytecodes;

CIL bytecodes; and

.NET bytecodes.

Claims 24, 39, and 54 have similar limitations to claim 9 and are rejected for similar reasons. The only differences are that claim 24 is for a method and claim 39 and 54 are for a computer program product.

Referring to claims 10, 25, 40, and 55, taking claim 10 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 6, wherein said non-native instructions are native program instructions of a different apparatus for processing data (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 25, 40, and 55 have similar limitations to claim 10 and are rejected for similar reasons. The only differences are that claim 25 is for a method and claims 40 and 55 are for a computer program product.

Referring to claims 11, 26, 41, and 56, taking claim 11 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 10, wherein said processing logic and said instruction decoder are part of a RISC processor and said non-native instructions are native instructions of a CISC processor (Smith Abstract; column 1, lines 14-20; column 3, lines 9-20). Claims 26, 41, and 56 have similar limitations to claim 11 and are rejected for similar reasons.

The only differences are that claim 26 is for a method and claims 41 and 56 are for a computer program product.

Referring to claim 14, 29, 44, and 59, taking claim 14 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 1, wherein said memory access instruction is a load instruction operable to load into a destination register specified by a destination register field within said load instruction a load value dependent upon a value read from a memory location specified by said base register value (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 29, 44, and 59 have similar limitations to claim 14 and are rejected for similar reasons. The only differences are that claim 29 is for a method and claims 44 and 59 are for a computer program product.

Referring to claims 15, 30, 45, and 60, taking claim 15 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 1, wherein said memory access instruction is a store instruction operable to store into a memory location specified by said base register value a store value dependent upon source value stored within a source register specified by a source register field within said store instruction (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 30, 45, and 60 have similar limitations to claim 15 and are rejected for similar reasons. The only differences are that claim 30 is for a method and claims 45 and 60 are for a computer program product.

Claims 2-5, 12-13, 17-20, 27-28, 32-35, 42-43, 47-50, and 57-58 are rejected under 35 U.S.C. 103(a) as being unpatentable over Click, Jr. et al., U.S. Patent Number 6,363,522 (herein referred to as Click) in view of Smith et al., U.S. Patent Number 5,430,862 (herein referred to as

Smith) as applied to claims 1, 16, 32, and 46 above, and further in view of Mirapuri et al., U.S. Patent Number 5,590,294 (herein referred to as Mirapuri).

Referring to claims 2, 17, 32, and 47, taking claim 2 as exemplary, Click in view of Smith has taught an apparatus as claimed in claim 1, but not taught wherein in response to said memory access instruction a return address is stored pointing a memory location storing a program instruction to be executed upon a return from said null value exception handler. Mirapuri has taught wherein in response to said memory access instruction a return address is stored pointing a memory location storing a program instruction to be executed upon a return from said null value exception handler (Mirapuri column 2, lines 41-51; column 7, lines 36-53; column 10, lines 15-41; column 12, line 53 to column 13, line 10; and Figure 8). A person of ordinary skill in the art at the time the invention was made, and as taught by Mirapuri, the device of Mirapuri improves pipeline throughput (Mirapuri column 3, lines 3-5), thereby improving processor efficiency and speed. Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the co-processing of Mirapuri in the device of Smith to improve processor efficiency and speed. Claims 17, 32, and 47 have similar limitations to claim 2 and are rejected for similar reasons. The only differences are that claim 17 is for a method and claims 32 and 47 are for a computer program product.

Referring to claims 3, 18, 33, and 48, taking claim 3 as exemplary, Click in view of Smith and in further view of Mirapuri have taught an apparatus as claimed in claim 1, wherein said null value exception handler is located at a memory address pointed to by a value stored within a programmable configuration register (Mirapuri column 2, lines 41-51; column 7, lines 36-53; column 10, lines 15-41; column 12, line 53 to column 13, line 10; and Figure 8). Claims

18, 33, and 48 have similar limitations to claim 3 and are rejected for similar reasons. The only differences are that claim 18 is for a method and claims 33 and 48 are for a computer program product.

Referring to claims 4, 19, 34, and 49, taking claim 4 as exemplary, Click in view of Smith and in further view of Mirapuri have taught an apparatus as claimed in claim 3, wherein said programmable configuration register is a coprocessor configuration register (Mirapuri column 2, lines 41-51; column 7, lines 36-53; column 10, lines 15-41; column 12, line 53 to column 13, line 10; and Figure 8). Claims 19, 34, and 49 have similar limitations to claim 19 and are rejected for similar reasons. The only differences are that claim 19 is for a method and claims 34 and 49 are for a computer program product.

Referring to claims 5, 20, 35, and 50, taking claim 5 as exemplary, Click in view of Smith and in further view of Mirapuri have taught an apparatus as claimed in claim 3, wherein said branch is made to an instruction stored at a memory address given by said value stored within said programmable configuration register subject to a fixed offset (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 20, 35, and 50 have similar limitations to claim 5 and are rejected for similar reasons. The only differences are that claim 20 is for a method and claims 35 and 50 are for a computer program product.

Referring to claims 12, 27, 42, and 57, taking claim 12 as exemplary, Click in view of Smith and in further view of Mirapuri have taught an apparatus as claimed in claim 3, wherein said value stored within said programmable configuration register is a start address of said null value exception handler (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 27, 42, and 57 have similar limitations to claim 12 and are rejected for similar reasons. The only differences are that claim 12 is for a method and claims 42 and 57 are for a computer program product.

Referring to claims 13, 28, 43, and 58, taking claim 13 as exemplary, Click in view of Smith and in further view of Mirapuri have taught an apparatus as claimed in claim 3, wherein said value stored within said programmable configuration register is an address of a jump instruction operable to jump execution to a start address of said null value exception handler (Click column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7). Claims 28, 43, and 58 have similar limitations to claim 13 and are rejected for similar reasons. The only differences are that claim 13 is for a method and claims 43 and 58 are for a computer program product.

The following table shows a clearer mapping of the claim limitations to cited prior art for the independent claims 1 and 46.

Instant Application	Prior Art
Claim 1	
An apparatus for processing data comprising:	
Processing logic operable to perform data processing operations; and	Smith Column 2, line 53 to column 4, line 43 "FIG. 1 shows in block diagram form, a data processing system..." Figure 1, element 10 Figure 2, elements 12-2 and 12-4 Click has not explicitly taught the processing logic or instruction decoder details. However, Click has taught the memory access instructions are executed on a processor, as shown in Click's Figure 5, but provides no details about the "N Processor". Smith has taught the details of the processor.

	A person of ordinary skill in the art at the time the invention was made, and as taught by Smith, would have recognized that the processor of Smith increases performance and compatibility by allowing instructions from multiple instruction sets execute (Smith column 1, lines 36-39) and reducing the need to access off-chip memory (Smith column 2, lines 13-24). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the processor of Smith in the device of Click to increase processor performance and compatibility.
An instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions;	Smith Column 2, line 53 to column 4, line 43 "...The E chip **12-2** performs three main functions...cracks or decodes each instruction...the unit **12-200** operates to fetch and decode each source instruction...Microprocessor chip **12-4** further includes...an instruction decode and control unit..." Figure 1, elements 12-2 and 12-4 Figure 2, elements 12-406 and 12-200
Wherein said instruction decoder, in response to a memory access instruction, compares a base register value, stored within a base register specified by a base register field of said memory access instruction, with a predetermined null value; and,	Click Column 1, line 66 to column 2, line 47 "...Source code **102** includes commands, or instructions, to load contents associated with pointers into registers..." Column 3, line 48 to column 4, line 7 "...determining whether the first code portion is redundant with respect to the second code portion includes determining if the first code portion is arranged to be reached by a program control flow associated with the source code..."
If said base register value matches said predetermined null value, then said decoder triggers branching to execution of a null value exception handler.	Click Column 1, line 66 to column 2, line 47 "...When an exception is thrown in response to the identification of a null pointer..." Column 3, line 48 to column 4, line 7 "...each of the first code potion and the second code portions includes a determination of whether to throw a particular exception..."
Claim 46	Prior Art
A computer program product	

comprising a computer readable storage medium containing	
Computer readable instructions for translating non-native program instructions to form native program instructions directly decodable by an apparatus for processing data having	Smith Column 2, line 53 to column 4, line 43 "...The E chip **12-2** performs three main functions. It fetches source instructions executable by a general purpose computer from instruction cache **18**, cracks or decodes each instruction and, based upon the result of such decoding, generates a number of branch vector addresses which specify the corresponding emulation subroutines for executing each source instruction...The microprocessor **12-4** performs the main function of executing the required emulation subroutines specified by the vector branch addresses generated by E chip **12-2**... the unit **12-200** operates to fetch and decode each source instruction ...Microprocessor chip **12-4** further includes...an instruction decode and control unit..." Figure 1, elements 12-2 and 12-4 Figure 2, elements 12-406 and 12-200 Click has not explicitly taught translating the non-native to native instructions, the processing logic, or instruction decoder details. However, Click has taught the memory access instructions are executed on a processor, as shown in Click's Figure 5, but provides no details about the "N Processor". Smith has taught the details of the processor. A person of ordinary skill in the art at the time the invention was made, and as taught by Smith, would have recognized that the processor of Smith increases performance and compatibility by allowing instructions from multiple instruction sets execute (Smith column 1, lines 36-39) and reducing the need to access off-chip memory (Smith column 2, lines 13-24). Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the processor of Smith in the device of Click to increase processor performance and compatibility.
Processing logic operable to perform data processing operations and	Smith Column 2, line 53 to column 4, line 43 "FIG. **1** shows in block diagram form, a data processing system..."

	Figure 1, element 10 Figure 2, elements 12-2 and 12-4
An instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions,	Smith Column 2, line 53 to column 4, line 43 "...The E chip **12-2** performs three main functions...cracks or decodes each instruction...the unit **12-200** operates to fetch and decode each source instruction...Microprocessor chip **12-4** further includes...an instruction decode and control unit..." Figure 1, elements 12-2 and 12-4 Figure 2, elements 12-406 and 12-200
Said native program instructions comprising, In response to a memory access instruction decodable by said instruction decoder to control said processing logic comparing a base register value, stored within a base register specified by a base register field of said memory access instruction, with a predetermined null value; and,	Click Column 1, line 66 to column 2, line 47 "...Source code **102** includes commands, or instructions, to load contents associated with pointers into registers..." Column 3, line 48 to column 4, line 7 "...determining whether the first code portion is redundant with respect to the second code portion includes determining if the first code portion is arranged to be reached by a program control flow associated with the source code..."
If said base register value matches said predetermined null value, then triggering branching to execution of a null value exception handler.	Click Column 1, line 66 to column 2, line 47 "...When an exception is thrown in response to the identification of a null pointer..." Column 3, line 48 to column 4, line 7 "...each of the first code potion and the second code portions includes a determination of whether to throw a particular exception..."

The following table shows how claims 16 and 31 have similar limitations to claim 1 and, therefore, rejected for similar reasons presented in the table above.

Claim 1	Claim 16	Claim 31
An apparatus for processing data comprising:	A method of processing data with an apparatus for processing data	A computer program product comprising a computer readable storage

	having	medium containing computer readable instructions for controlling an apparatus for processing data having
Processing logic operable to perform data processing operations; and	Processing logic operable to perform data processing operations and	Processing logic operable to perform data processing operations and
An instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions;	An instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions,	An instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions,
Wherein said instruction decoder, in response to a memory access instruction, compares a base register value, stored within a base register specified by a base register field of said memory access instruction, with a predetermined null value; and,	Said method comprising the steps of: In response to said memory access instruction decoded by said instruction decoder controlling said processing logic comparing a base register value, stored within a base register specified by a base register field of said memory access instruction, with a predetermined null value; and,	Said computer program comprising, In response to a memory access instruction decodable by said instruction decoder to control said processing logic, comparing a base register value, stored within a base register specified by a base register field of said memory access instruction, with a predetermined null value; and,
If said base register value matches said predetermined null value, then said decoder triggers branching to execution of a null value exception handler.	If said base register value matches said predetermined null value, then triggering branching to execution of a null value exception handler.	If said base register value matches said predetermined null value, then branching to execution of a null value exception handler.

(10) Response to Argument

Examiner notes that Appellants' arguments on pages 8-10 with regards to the withdrawn 35 USC §112, first and second paragraph rejections are moot since the rejections have been withdrawn.

Appellants' argue in essence on page 11

> In view of the Examiner's admissions, Click cannot possibly teach the claimed interrelationship between the "processing logic" and the "instruction decoder" structures given that the structures aren't present in Click.

This has not been found persuasive. As an initial matter, the examiner submits that he cannot make an admission as to what the prior does or does not disclose. Moreover, appellant has mischaracterized the statement of the rejection. In the rejection copied above, it is stated that Click does not "explicitly" teach the detailed functionality of the processing logic and instruction decoder. Smith was relied upon to teach those details. However, Click has taught that a processor, which typically comprises processing logic and a decoder, is used to execute the instructions recited by other limitations in the claim. Smith was relied upon to teach the specific processing logic and instruction decoder recited in the claims. As shown in the rejection above, Click explicitly teaches in column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7 a processor executing instructions "wherein said instruction decoder, in response to a memory access instruction, compares a base register value, stored within a base register specified by a base register field of said memory access instruction, with a predetermined null value; and, if said base register value matches said predetermined null value, then said decoder triggers branching to execution of a null value exception handler." However, Click has not taught the specific details of the processor, such as "processing logic operable to perform data processing operations and an instruction decoder operable to decode program instructions to control said processing logic to perform data processing operations specified by said program instructions". Smith has taught a processor with these specific details in column 2, line 53 to column 4, line 43; Figure 1; and Figure 2. As supported by the Supreme Court decision on KSR Int'l. Co., v. Teleflex, Inc., 550 U.S.-, 82 USPQ2d 1385 (2007) (herein referred to as KSR) as to the elements for the test of obviousness, the Examiner identified what the prior art, in this case Click, has taught, which is, in summary, determining whether there is an exception during a memory instruction by comparing a base pointer to a null value and, when an exception is present, triggering branching to an exception handler. The Examiner then identified what is not taught explicitly in Click, which is the specific structure of the processor, including the processing logic and the decoder. The Examiner then found secondary prior art, in this case Smith, which has taught these missing elements.

Appellant argues in essence on page 12

> In citing the Smith reference the Examiner make only a general allegation that Smith teaches the "processing logic" and the "instruction decoder" which he admits is missing from Click reference. However, the Examiner disregards the details of the "instruction decoder" which are specified in the last portion of the independent claims...

This has not been found persuasive. As presented in the previous rejection and copied above, Click in column 1, line 66 to column 2, line 47 and column 3, line 48 to column 4, line 7 teaches the functionality described in the last portion of the claims. While Click taught a processor performing the functionality, Click did not explicitly teach processing logic or an instruction decoder in the processor, so the rejection relied upon Smith to teach the specific processing logic and instruction decoder. In column 2, line 53 to column 4, line 43; Figure 1; and Figure 2, Smith has taught a processor with an emulation portion and a microprocessor portion. The emulation portion fetches instruction, decodes them, and, based upon the decoding, generates branch vector addresses that correspond to subroutines for executing the source instruction. The microprocessor portion executes the subroutines specified by the vector branch addresses generated by the emulation portion. As supported by KSR as to the elements for the test of obviousness, the Examiner identified what the prior art, in this case Click, has taught, which is, in summary, determining whether there is an exception during a memory instruction by comparing a base pointer to a null value and, when an exception is present, triggering branching to an exception handler. The Examiner then identified what appears to be lacking in Click, which is the specific structure of the processor, including the processing logic and the decoder. The Examiner then found secondary prior art, in this case Smith, which has taught these missing elements.

Appellant argues in essence on pages 12-13 and 16-18

> ...The Examiner has provided no evidence of record establishing any "reason" or "motivation"...

This has not been found persuasive. As supported by KSR, the motivation to combine references need not have explicit evidence to support it and may be based on reasoning as simple as "obvious to try" and common sense. In the combination of Click in view of Smith, since Click has not provided the details of the processor performing the functions claimed for determining whether there is an exception during a memory instruction by comparing a base pointer to a null value and, when an exception is present, triggering branching to an exception handler. Smith provides a known processor layout that determines whether an instruction needs to be executed via a subroutine, provides the subroutine branch address if it does, and executes the subroutine at the indicated subroutine branch address. A person of ordinary skill in the art would have recognized that the functionality of Click could easily be performed by the processor of Smith, since the instruction would be an exception instruction, the subroutine branch address would be the exception handler address, and the executed subroutine would be the triggered exception handler. In addition, the Examiner has provided specific reasons stated by Smith himself as the benefits of his processor, including increasing performance and compatibility by allowing instructions from multiple instruction sets execute (Smith column 1, lines 36-39) and reducing the need to access off-chip memory (Smith column 2, lines 13-24). In regards to Mirapuri, again, the Examiner explicitly stated in the rejection that Mirapuri "improves pipeline throughput (Mirapuri column 3, lines 3-5), thereby improving processor efficiency and speed" as the motivation to combine the references.

Appellant argues in essence on pages 13-16

> ...The Click and Smith references are mutually incompatible and therefore cannot and would not be combined as suggested by the Examiner...

This has not been found persuasive. The Examiner attempted in the Final Rejection to illustrate that whether Click and Smith were implemented in hardware or software was irrelevant to their compatibility, since the choice of hardware or software implementation is a design choice, and attempted to show support for this assertion by citing Tanenbaum. However, Appellants' arguments appear to assume that Tanenbaum was meant to be combined into the rejection as a third reference, when it was meant as supportive evidence against Appellants' assertion that, since Click teaches software and Smith teaches hardware, they are incompatible. Tanenbaum clearly states in the cited portions that whether a portion of the design is implemented in hardware or software is a design decision. Appellants' arguments focus upon the fact that "Click relies upon an analysis made during compilation to remove redundant null checks from the code as it is being compiled from source code form into runtime bytecodes" and Smith "discusses use of separate translating programs (such as compilars) to convert source computer programs to an executable form...then dismisses this approach as too time consuming and inefficient." Click is referring to compilers which translate higher level programming languages into machine code recognized by the processor, such as JAVA™ into machine assembly language, while Smith refers to translation programs that simulate or emulate operations of different types of computers, i.e. two different machine assembly languages. Smith still needs a compiler to translate higher level languages into machine language to function properly, because, without a compiler translating the higher level language programs, the processor would not know what instructions to execute.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Aimee J. Li

Aimee J. Li
AU 2183 Patent Examiner

Conferees:

[illegible]
EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Eddie Chan
AU 2183 Supervisory Patent Examiner

/Lynne H Browne/
Lynne H. Browne
Appeal Practice Specialist, TQAS
Technology Center 2100